SINDH BOARD NOTES

COMPUTER SCIENCE

10 TH CLASS

Presented by:

Urdu Books Whatsapp Group

STUDY GROUP

0333-8033313
راؤ ایاز

0343-7008883
پاکستان زندہ باد

0306-7163117
محمد سلمان سلیم

Unit 1: Introduction to Programming

Exercise Solutions

Q1. Multiple Choice Questions

1. A software that facilitates programmers in writing computer programs is known as _____.

a. compiler
b. An Editor
c. An IDE
d. A debugger

2. _______ is a software that is responsible for the conversion of program files to machine-understandable and executable code.

a. Compiler
b. Editor
c. IDE
d. Debugger

3. Every programming language has some primitive building blocks and follows some grammar rules known as its_____.

a. Programming rules
b. Syntax
c. Building blocks
d. Semantic rules

4. A list of words that are predefined and must not be used by the programmer to name his own variables are known as _____.

a. Auto words
b. Reserved words
c. Restricted words
d. Predefined words

5. Include statements are written in ______sections.

a. Header
b. Main
c. Comments
d. Print

6. ______are added in the source code to further explain the techniques and algorithms used by the programmer.

a. Messages
b. Hints
c. Comments
d. Explanations

7. ______are the values that do not change during the whole execution of program.

a. Variables
b. Constants
c. Strings
d. Comments

8. A float uses ______ bytes of memory.
a. 3
b.4
c.5
d. 6

9. For initializing a variable, we use _______ operator.
a. →
b. =
c. @
d. ?

10. ______ can be thought of as a container to store constants.
a. Box
b. Jar
c. Variable
d. Collection

Answers

1.c	2.a	3.b	4.b	5.a	6.c	7.b	8.b	9.b	10.c

Q2. True of False
1. An IDE combines text editors, libraries, compilers and debuggers in a single interface.
2. Computer require the conversion of the code written in program file to machine language in order to execute it.
3. Column is a reserved word in C programming language.
4.*comment goes here* is a valid comment.
5. float can store a real number up to six digits of precision.

Answers
1. T
2. T
3. F
4. F
5. T

Q3. Define the following:
1) IDE
A software that provides a programming environment for programmers to write and execute computer programs is known as **Integrated Development Environment (IDE).** The user can interact with the IDE using windows and buttons to provide input and get output.
2) Compiler

A compiler is a program that converts the instructions of a high level language into machine language. The compiler also detects syntax error. It does not compile a program that contains a syntax error

3) Reserved Words

Reserved word has a predefined meaning and purpose in a programming language. The meaning and purpose of a reserved word is defined by the developer of the language. It can be used for the same purpose for which it is defined.

4) Main section of program

The **main()** function is the place where execution of C program starts. When a program executes, the control enters **main()** function and starts executing its statements. Every C program must contain a **main()** function.

5) char data type

A variable of character data type can store one character. A single character may be an alphabet, a digit or a special character. The character value is normally given in single quotes such as 'a', 'x', '5', and '#'. The char keyword is used to declare a variable of character data type. It takes 1 byte in the memory.

Q4. Briefly answer the following questions:

1. Why do we need a programming environment?

We need programming environment to write and execute computer programs. It is important to setup the programming environment before writing the programs.

2. Write the steps to create a C program file in the IDE of your lab computer.

The steps to create a C program are as follows:

1. Start **Code Blocks IDE.**
2. Click **File > New > Empty File.** A new empty file will appear in the IDE.
3. Type the desired program code.
4. Click **File > Save File** or press **Ctrl+S**. The **Save** file dialog box will appear.
5. Type the desired filename in **File** name box and click **Save.** The C program file will be created with the given name.

3. Describe the purpose of a compiler.

The computer cannot understand a program written in high-level language directly. The purpose of a compiler is to translate the instructions written in high-level languages into machine language for the computer to understand and execute.

4. List down five reserved words in C programming language.

int, case, return, for, if

5. Discuss the main parts of the structure of C program.

The main parts of the structure of C program include header section, main section and body of main() function. The header section is the part where header files are included. The main section refers to the main() function. The body of main() function contains all the statements enclosed in curly braces.

6. Why do we use comments in programming?

ختم نبوت ﷺ زندہ باد　　　　　　　　　　عظمت صحابہ زندہ باد

السلام علیکم ورحمۃ اللہ وبرکاتہ:

معزز ممبران: آپ کا وٹس ایپ گروپ ایڈ من "**اردو بکس**" آپ سے مخاطب ہے۔

آپ تمام ممبران سے گزارش ہے کہ:

- گروپ میں صرف PDF کتب پوسٹ کی جاتی ہیں لہذا کتب کے متعلق اپنے کمنٹس / ریویوز ضرور دیں۔ گروپ میں بغیر ایڈ من کی اجازت کے کسی بھی قسم کی (اسلامی وغیر اسلامی، اخلاقی، تحریری) پوسٹ کرنا سختی سے منع ہے۔
- گروپ میں معزز، پڑھے لکھے، سلجھے ہوئے ممبرز موجود ہیں اخلاقیات کی پابندی کریں اور گروپ رولز کو فالو کریں بصورت دیگر معزز ممبرز کی بہتری کی خاطر ریموو کر دیا جائے گا۔
- کوئی بھی ممبر کسی بھی ممبر کو انباکس میں میسیج، مس کال، کال نہیں کرے گا۔ رپورٹ پر فوری ریموو کر کے کاروائی عمل میں لائے جائے گی۔
- ہمارے کسی بھی گروپ میں سیاسی و فرقہ واریت کی بحث کی قطعاً کوئی گنجائش نہیں ہے۔
- اگر کسی کو بھی گروپ کے متعلق کسی قسم کی شکایت یا تجویز کی صورت میں ایڈ من سے رابطہ کیجئے۔
- سب سے اہم بات:

گروپ میں کسی بھی قادیانی، مرزائی، احمدی، گستاخِ رسول، گستاخِ امہات المؤمنین، گستاخِ صحابہ و خلفائے راشدین حضرت ابو بکر صدیق، حضرت عمر فاروق، حضرت عثمان غنی، حضرت علی المرتضیٰ، حضرت حسنین کریمین رضوان اللہ تعالیٰ اجمعین، گستاخ اہلبیت یا ایسے غیر مسلم جو اسلام اور پاکستان کے خلاف پراپیگنڈا میں مصروف ہیں یا ان کے روحانی وذہنی سپورٹرز کے لئے کوئی گنجائش نہیں ہے لہذا ایسے اشخاص بالکل بھی گروپ جوائن کرنے کی زحمت نہ کریں۔ معلوم ہونے پر فوراً ریموو کر دیا جائے گا۔

- تمام کتب انٹرنیٹ سے تلاش / ڈاؤنلوڈ کر کے **فری آف کاسٹ** وٹس ایپ گروپ میں شیئر کی جاتی ہیں۔ جو کتاب نہیں ملتی اس کے لئے معذرت کر لی جاتی ہے۔ جس میں محنت بھی صرف ہوتی ہے لیکن ہمیں آپ سے صرف دعاؤں کی درخواست ہے۔
- عمران سیریز کے شوقین کیلئے علیحدہ سے عمران سیریز گروپ موجود ہے۔
- **لیڈیز کے لئے الگ گروپ کی سہولت موجود ہے جس کے لئے ویریفکیشن ضروری ہے۔**
- اردو کتب / عمران سیریز یا سٹڈی گروپ میں ایڈ ہونے کے لئے ایڈ من سے وٹس ایپ پر بذریعہ میسیج رابطہ کریں اور جواب کا انتظار فرمائیں۔ برائے مہربانی اخلاقیات کا خیال رکھتے ہوئے موبائل پر کال یا ایم ایس کرنے کی کوشش ہر گز نہ کریں۔ ورنہ گروپس سے تو ریموو کیا ہی جائے گا بلاک بھی کیا جائے گا۔

نوٹ: ہمارے کسی گروپ کی کوئی فیس نہیں ہے۔ سب فی سبیل اللہ ہے

0306-7163117	0343-7008883	0333-8033313
محمد سلمان سلیم	پاکستان زندہ باد	راؤ ایاز

پاکستان زندہ باد　　　　　　　　　　پاکستان پائندہ باد

اللہ تبارک تعالیٰ ہم سب کا حامی وناصر ہو

Comments are used to help other programmers in understanding the code easily. Comments are notes about the code that explain its purpose. It helps in understanding the program even after a long time,

7. Differentiate between constants and variables.

The variable is a named memory location whereas the constant is a named value. The value of constant can change during program execution but the value of constant cannot change during program execution

8. Write down the rules for naming variables.

Each variable must have a unique name or identifier. The variable name must be meaningful and readable. It helps the user to understand the purpose of the variable. For example, a variable to store the marks of student should be marks.

Different rules for naming variables in C language are as follows:

1. Variable may include letters, numbers and underscore (_).

2. The first character of variable must be a letter or underscore_. It cannot begin with a digit. The variables 9minute, #home and 2kg are invalid.

3. Blank spaces are not allowed in variable names. For example, my var and your car are invalid

4. The special symbols such as @, $, % are not allowed in variable names. For example, $cost and tot.al are invalid variable names.

5. Reserved word cannot be used as variable name. For example, int, void and while are invalid variables,

6. A variable can be up to 31 characters long for many compilers. If a variable consists of more than 31 characters only first 31 characters will be used. The remaining characters will be ignored.

Examples

Some examples of valid variables are as follows:

Marks   sub1   _test   f_name

9. Differentiate between char and int.

int stands for Integer data type and allows a variable to store numeric values. It takes 4 bytes of memory. **char** stands for character data type and allows a variable to store only one character. It takes 1 byte of memory.

10. How can we declare and initialize the variable?

A variable can be declared and initialized as follows:

int age = 15;

Match the Columns

A	B	C(Ans)
1) IDE	a) Machine executable code	d
2) Text Editor	b) include statement	f
3) Compiler	c) python	a
4) Programming Language	d) CLion	c
5) Reserved words	e) /* (a+b) */	g

6) Link Section	f) Notepad	b
7) Body of main ()	g) struct	h
8) Comment	h) { }	e

Unit 2: User Interaction

Exercise Solutions

Q1. Multiple Choice Questions

1. printf is used to print ________ type of data.

a. int b. float c. char d. All

2. scanf is a ________ in C programming language.

a. Keyword b. Library c. Function d. None

3. getch() is used to take ________ as input from user.

a. int b. float c. char d. All

4. Let the following part of code, what will be the value of variable a after execution?

```
int a = 4;
float b = 2.2;
a = a * b;
```

a. 8.8 b. 8 c. 8.0 d. 8.2

5. Which of the following is a valid line of code?

a. int = 20;
b. grad = 'A'
c. line = this is a line;
d. None of them

6. Which operator has highest precedence among the following?

a. / b. = c. > d. !

7. Which of the following is not a type of operator?

a. Arithmetic operator
b. Relational operator
c. Check operator
d. Logical operator

8. The operator % is used to calculate ________.

a. Percentage
b. Remainder
c. Factorial
d. Square

9. Which of the following is a valid character?

a. 'here' b. "a" c. '9' d. None

10. What is true about C language?

a. C is not a case sensitive language
b. Keywords can be used as variable names
c. All logical operators are binary operators
d. None of them

Answers

1. d	**2. c**	**3. c**	**4. b**	**5. b**
6. d	**7. c**	**8. b**	**9. c**	**10. d**

Q2. True of False

1. Maximum value that can be stored by an integer is 32000
2. Format specifiers begin with a % sign.
3. Precedence of division operator is greater than multiplication operator.
4. getch() is used to take all types of data input from user.

5. scanf() is used for output operations.

Answers

1. F **2. T** **3. F** **4. F** **5. F**

Q3. Define the following:

1) Statement terminator

Statement Terminator

The semicolon is used as a **statement terminator** in C programs. It indicates the end of a statement. Every statement in C program is terminated with a semicolon (;). The compiler generates a syntax error if a statement is not terminated by semicolon.

An example of using statement terminator is as follows:

int n = 50;

2) Format specifier

Format Specifier

Format specifier is used to specify the format of data during input and output operations. A format specifier starts with the symbol %. Format specifiers can be used with variables, constants and expressions.

Different format specifiers are used for different data types as follows:

Data type	**Format Specifier**
int	%d or %i
float	%f
char	%c

Table: Format specifier for input/output

3) Escape sequence

Escape Sequences

Escape sequences are special characters used to modify the format of output. They are used in format string of **printf** function. These characters are not displayed in the output. They change the normal working of **printf** function to display the output.

Some commonly used escape sequences in C language are as follows:

Escape Sequence	**Purpose**
\n	New line
\t	Tab
\’	Single quote
\"	Double quote
\\	Backslash

\b	Backspace
\a	Alert

Table: Escape sequences and their purpose

4) scanf

scanf Function

The scanf function is used to get input from the user. The input is stored in a variable in a specified form using format specifier. For example, the format specifier %d or %i must be used in scanf function to input integer data.

Syntax

The syntax of scanf function is as follows:

scanf(format string, &variablel , &variable2, &variable3 ...);

5) Modulus operator

Modulus Operator

The **modulus operator** gives the remainder of division of two integers. It is also called **remainder operator**. It only works with integer values.

Example

int n = 33 % 6;

The variable n contains the value 3 after executing the above statement.

Q.4. Briefly answer the following questions:

1. What is difference between scanf and getch?

Ans: The scanf() function is used to get different types of inputs from the user. The user needs to press **Enter** after typing the input value. The getch() function is used to input single character. The user does not need to press **Enter** after typing the input value.

2. Which function of C language is used to display output on screen?

Ans: The **printf()** function of C language is used to display output on screen. It can display text, constants or values of variables.

3. Why format specifiers are important to be specified in I/O operations?

Ans: The format specifiers are important to be specified in I/O operations. They tell the compiler what type of data is in a variable during taking input using **scanf()** or printing using **printf().**

4. What are escape sequences? Why do we need them?

Ans: Escape sequence consists of two characters. The first character is always backslash. The second character varies according to desired functionality. Escape sequences are needed to modify the format of output. They are used in format string of **printf** function. They change the normal working of printf function to display the output.

5. Which operators are used for arithmetic operations?

Ans: Arithmetic operators are used to perform arithmetic operations. C provides different arithmetic operators such as addition (+), subtraction (-), multiplication (*), division (/) and modulus operator (%).

6. What are relational operators? Describe with an example.

Ans: The relational operators compare two values to determine the relationship between these values. They perform operations on two operands and produce the result as true or false. For example, the greater than operator (>) determines whether one value is greater than another. The expression **12 > 5** will return true.

7. What are logical operators? Describe with an example.

Ans: The logical operators perform operations on relational expressions and produce the result as true or false. They can be used to evaluate more than one relational expressions. For example, the expression 9 > 5 && 50 < 100 will return true.

8. What is the difference between unary and binary operator?

Ans: The operators that work with single operand are called unary operator whereas the operators that work with two operands are called binary operator. Some unary operators include -, ++, - - and ! (NOT). Some binary operators include +, -, *, /, %, && and || .

9. What is the difference between = = operator and = operator?

Ans: The operator = is the assignment operator. It is used to assign the result of an expression on the right side to the variable on the left side. The operator = = is a relational operator. It is used to compare two values to determine if they are equal or not.

10. What is meant by precedence of operators? Which operator has highest precedence in C language?

Ans: The order in which different types of operators in an expression are evaluated is known as operator precedence. If an expression has different types of operators, the operators with higher precedence are evaluated before the operators with lower precedence. The logical NOT (!) has the highest precedence In C language.

Q5. Write down the output of the following code segment.

a.

```
#include <stdio.h>
void main()
{
    int x=2, y=3, z=6;
    int ansl , ans2, ans3;
    ans1 = x / z * y;
    ans2= y + z / y * 2;
    ans3= z / x + x * y;
    printf("%d %d %d", ansl , ans2, ans3);
```

```
}
```

Output:

0 7 9

b.

```
#include <stdio.h>
void main()
{
        printf( "nn \n\n nnn\nn\nt\t" );
        printf( "nn /n/n nn/n\n" );
}
```

Output:

```
nn

 nnn
n
t       nn /n/n nn/n
```


c.

```
#include <stdio.h>
void main()
{
        int a = 4, b;
        float c = 2.3;
        b = c * a;
        printf("%d", b);
}
```

Output:

9

d.

```
#include <stdio.h>
void main()
{
        int a = 4 * 3 / (5+1) + 7 % 4;
        printf("%d ", a);
}
```

Output:

5

e.

```
#include <stdio.h>
void main()
{
        printf("%d",( ((5>3) && (4>6)) || (7>3) ));
}
```

Output:
1

Q6. Identify errors in the following coding segments.

a.

```
#include <stdio.h>
void main()
{
        int a, b = 13;
        b = a % 2;
        printf("Value of b is : %d, b);
}
```

Ans: The closing double quote is missing after %d in printf function in line 6.

b.

```
#include <stdio.h>
void main()
{
        int a, b, c,
        printf("Enter First Number: " );
        scanf("%d", & a);
        printf("Enter Second Number: ");
        scanf("%d", & b);
        a+b=c;
]
```

Ans:

1. The variable declaration in line 4 must end with semicolon instead of comma.
2. The line 9 must be written as c = a + b;
3. The closing bracket] in the last line must be replaced with }.

c.

```
#include <stdio.h>;
main ()
{
        int num;
```

```
        printf(Enter number: ");
        scanf(%d , &num);
};
```

Ans:

1. The semicolon at the end of line 1 is invalid.
2. The return type void is missing in line 2.
3. The starting double quote is missing in printf function in line 5.
4. The format specifier %d must be enclosed in double quotes in line 6.
5. The semicolon after } in the last line is invalid.

d.

```
include <stdio.h>
int main()
{
        float f;
        printf["Enter value: "];
        scanf("%c", &f);
}
```

Ans:

1. The line 1 must start with #.
2. The brackets [ ] in line 5 must be replaced with parenthesis ( ).
3. The format specifier %c in line 6 must be replaced with %f.

Unit 3: Conditional Logic

Exercise Solutions

Q1. Multiple Choice Questions

1) Conditional logic helps in

a. Decision b. Iterations c. Traversing d. All

2) _________ statement describe the sequence in which statements of program should be executed.

a. Loop b. Conditional c. Control d. All

3) In if statement, what happens if condition is false?

a. Program crashes b. Index out of bound error
c. Further code executes d. Compiler asks to change condition

4)

```
int a = 5;
if (a < 10)
    a++;
else
    if (a > 4)
    a--;
```

Which of the following statements will execute?

a. a++ b. a-- c. Both a and b d. None

5) Which of the following is the condition to check a is a factor of c?

a. a % c == 0; b. c % a == 0;
c. a * c == 0; d. a + c == 0;

6) A condition can be any __________ expression.

a. Arithmetic b. Relational
c. Logical d. Arithmetic, relational or logical

7) An if statement inside another if statement is called _________ structure.

a. Nested b. Boxed
c. Repeated d. Decomposed

8) A set of multiple instructions enclosed in braces is called a __________.

a. Box b. List c. Block d. Job

Answers

1. a	**2. d**	**3. c**	**4. a**
5. b	**6. d**	**7. a**	**8. c**

Q.2. Define the following.

<u>1) Control Statements</u>

Control Statement

A statement used to control the flow of execution in a program is called **control statement**. The flow of execution is controlled in different ways to solve a problem. For example, sometimes one set of statements is executed if a condition is **true**. The other set of statements is executed when the condition is **false**. Similarly, some set of statements needs to be executed repeatedly for a number of times.

2) Selection Statements

Selection Control Statement

The **selection control statement** executes a statement or set of statements based on a condition. It executes a statement if the given condition is **true** and ignores when the condition is **false**.

3) Sequential Control

1. Sequential Control Statement

The **sequential control statement** executes the statements in the same order in which they are written in the program. It is the default control structure in C language. The statements are executed one after the other. All statements of the program are executed.

4) Condition

Condition

A **condition** is an expression that is either **true** or **false**. An expression that gives a non-zero value is considered **true**. An expression that gives 0 value is considered **false**. The condition may consist of arithmetic expressions, relational expressions and logical expressions.

Examples

Some examples of valid expressions that can be used as conditions are as follows:

Expression	Result
5 > 3	True
!(10 > 5)	False

5) Nested Selection Structures

Nested Selection Structure

A selection structure within a selection structure is called **nested selection structure**. The **nested if** statement is a nested selection structure. In nested structure, the control enters into the **inner if** only when the outer condition is **true**. Only one block of statements are executed and remaining blocks are skipped automatically.

Q3. Briefly answer the following questions:

1. Why do we need selection statements?

Ans: We need **selection control statements** to make decisions in programming based on a condition. They execute a statement if the given condition is **true** and ignore when the condition is **false**. **For example**, a selection control statement may display **"Pass"** if the marks of a student are 33 or more.

2. Differentiate between sequential and selection statements.

Ans: The **sequential statements** are executed in the same order in which they are written in the program. They are never skipped or repeated. The **selection statements** are executed based on a condition. They are only executed if the given condition is **true** and are skipped when the condition is **false**.

3. Differentiate between if statement and if-else statement with an example.

Ans: **if** statement is used to execute or skip a statement or set of statements based on a condition. The statement is only executed if the condition is **true**. Otherwise it is skipped. For example:

```
if (marks >= 60)
        printf("You passed.");
```

if-else statement is used for making two-way decisions. It executes one block of statement(s) when the condition is **true** and the other when it is **false**. For example:

```
if (marks >= 60)
        printf("You passed.");
else
        printf("You did not pass.");
```

4. What is the use of nested selection structures?

Ans: Nested selection structures are used when more than one decision must be made before carrying out a task. Many complex logic problems can be solved by using nested control structures.

5. Write the structure of if statement with brief description.

Ans: The structure of if statement is as follows:

```
if (condition)
        statement;
```

In the above structure, **if** is a keyword that is followed by a **condition** within the parentheses ( ). A **condition** can be any valid expression that evaluates to **true** or **false**. The statement after **if** condition is executed only if the result of **condition** is **true**.

Q4. Identify errors in the following coding segments. Assume that variables have already been declared.

a.

```
if ( x > 10 )
        printf( " Good ");
}
```

Ans: The opening braces { is missing.

b.

```
if ( a<b && b<c);
        sum = a + b + c;
else
        multiply = a * b * c;
```

Ans: The semicolon at the end of the first line is invalid.

c.

```
if( a<7<b )
        printf(“7");
```

Ans: The condition is invalid and must be written as if (a<7 && 7<b).

d.

```
if (a == b &| x == y)
        flag = true ;
else
        flag = false;
```

Ans: 1. The operator &| is invalid and must be written as && or || .
2. The values true and false are invalid.

e.

```
if (sum == 60 || product == 175 )
        printf( "Accepted %c), sum ) ;
else
        if (sum >= 45 || product > 100)
                printf ("Considered %d " + sum) ;
else
        printf ("Rejected" ) ;
```

Ans: 1. The closing parenthesis) after %c must be replaced with double quote " in line 2.
2. The plus sign + must be replaced with comma in line 5.

Q5. Write down the output of the following code segment.

a.

```
int a= 7, b= 10 ;
a = a + b ;
if ( a > 20 && b < 20 )
        b = a + b ;
printf (“ a = %d, b = %d “, a, b );
```

Ans: **a = 17, b = 10**

b.

```
int x= 45 ;
if ( x + 20 * 7 == 455 )
        printf( " Look's Good " );
else
        printf( " Hope for the Best " );
```

Ans: Hope for the Best

c.

```
char cl = 'Y' , c2 = 'N' ;
int n1 = 5, n2 = 9;
n1 = n1 + 1;
cl = c2;
if ( n1 == n2 && cl == c2)
        printf("%d = %d and %c = %c ", nl, n2, cl, cl);
else if ( n1 < n2 && cl == c2)
        printf( " %d < %d and %c = %c ", nl, n2, c1, c2);
else
        printf("Better Luck Next Time!" );
```

Ans: 6 < 9 and N = N

d.

```
int a = 34, b = 32, c = 7, d = 15;
a = b + c + d;
if ( a < 100 )
        a = a * 2;
        b = b * c;
        c = c + d;
if ( a > b && c == d )
{
        c = d;
        b = c;
        a = b;
}
else
if ( a>b && c>d || b >= d + c)
{
        d = c * c;
        a = b * b ;
}
printf( " a = %d, b = %d, c = %d, d = %d ", a, b, c, d);
}
```

Ans: a = 50176, b = 224, c = 22, d = 484

e.

```
int x = 5, y = 7, z = 9;
if ( x % 2 == 0 )
        x ++ ;
else
        x = y + z;
printf( " x = %d \n", x);
if ( x % 2 == 1 && y % 2 == 1 && z % 2 == 1)
        printf (" All are Odd " );
if ( x > y || x < z )
{
        if ( x > y )
                y++;
        else
                if ( x < z )
                        x++ ;
}
printf ("x = %d, y= %d, z = %d ", x, y, z );
```

Ans: **x = 16**

x=16, y = 8, z = 9

Unit 4: Data and Repetition

Exercise Solutions

Q1. Multiple Choice Questions

1) An array is a ________ structure.
a. Loop b. Control c. Data d. Conditional

2) Array elements are stored at ______ memory location.
a. Contiguous b. Scattered c. Divided d. None

3) If the size of an array is 100, the range of indexes will be _______ .
a. 0-99 b. 0-100 c. 1-100 d. 2-102

4) _______ structure allows repetition of a set of instructions.
a. Loop b. Conditional c. Control d. Data

5) ______ is the unique identifier, used to refer to the array.
a. Data Type b. Array name c. Array size d. None

6) Array can be initialized ________ declaration.
a. At the time of b. After
c. Before d. Both a& b

7) Using loops, inside loops is called ________ loops.
a. for b. while c. do-while d. Nested

8) _______ part of for loop is executed first.
a. Condition b. Body
c. Initialization d. Increment / Decrement

9) ________ make it easier to read and write values in array.
a. Loops b. Conditions
c. Expressions d. Functions

10) To initialize the array in a single statement, initialize it _________ declaration.
a. At the time of b. After
c. Before d. Both a & b

Answers

1. c	**2. a**	**3. a**	**4. a**	**5. b**
6. d	**7. d**	**8. c**	**9. a**	**10. a**

Q2. Define the following terms.

<u>1) Data structure</u>

Data Structure

Data structure is a container to store the collection of data items in a specific layout. C language provides different data structures. Array is one of the most commonly used data structures in C.

2) Array

Array

An **array** is a data structure that can store multiple values of same data type. For example, an array of data type **int** can store integer values and an array of data type **float** can store real values. All values in an array are stored at consecutive memory locations.

3) Array Initialization

Array Initialization

The process of assigning values to array elements for the first time is called **array initialization**. An array can be initialized at the time of array declaration or later. The initialization process provides a list of values for array elements.

4) Loop structure

Loop

The control structure that executes a statement or set of statements repeatedly is called **loop**. Loops are also known as **iterative** or **repetitive** structure.

5) Nested Loop

Nested Loop

A loop within a loop is called **nested loop**. In nested loops, the inner loop is executed completely with each change in the value of counter variable of outer loop. The nesting can be done up to any level. The increase in level of nesting increases the complexity of nested loop.

Q3. Briefly answer the following questions:

1. Is loop a data structure?

Ans: No, loop is not a data structure. It is a repetitive structure that is used to repeat statement or set of statements for specific number of times.

2. What is the use of nested loops?

Ans: The nested loops are used when we need to repeat a loop. Suppose a loop displays the counting from 1 to 5 and we need to display the same counting for 3 times. Another loop will be used to repeat this loop. It will become a nested loop.

3. What is the advantage of initializing an array at the time of declaration?

Ans: The advantage of initializing an array at the time of declaration is that all array elements can be initialized at once. Otherwise, the user has to access each array element individually to initialize the array that becomes a time-consuming process.

4. Describe the structure of for loop.

Ans: The structure of for loop consists of initialization, condition, increment/decrement and loop body. The initialization is used to set the initial value of the counter variable. The condition is checked every time before executing the loop body. The increment / decrement part is executed after executing the loop body.

5. How can you declare an array? Briefly describe the three parts of array declaration.

Ans: An array is declared by specifying its data type, name and size. The data type indicates the data types of the values to be stored in the array. The name indicates the name of the array. The size indicates the total number of elements that can be stored in the array. It must be a constant value.

Q6. Identify errors in the following coding segments.

a.

```
int a[] = ({2},{3},{4});
```

Ans: The array declaration is invalid and must be written as **int a[] = {2, 3, 4};**

b.

```
for (int i = 0, i < 10, i++)
        printf("%d\n", i);
```

Ans: The commas in the first line must be replaced with semicolons.

c.

```
int a[] = {1,2,3,4,5};
for (int j=0, j < 5; j++)
        printf("%d", a(j));
```

Ans: **1.** The comma after j=0 in the second line must be replaced with semicolon.
2. The statement **a(j)** in the third line must be written as **a[j].**

d.

```
float f [] = { 1.4, 3.5, 7.3, 5.9 };
int size = 4;
for (int n = -1; n < size; n -- )
        printf("%f\n", f[n]);
```

Ans: The loop has a logical error and it becomes an infinite loop and must be written as: **for(int n=0; n<size; n++).**

e.

```
int count = 0;
for (int i=4; i < 6; i--)
        for (int j =i, j< 45; j ++)
        {
                count ++;
                printf("%count" , count) ;
```

```
        }
```

Ans: **1.** The outer loop is an infinite loop and **i--** must be replaced with **i++** to remove the logical error.

2. The comma in the inner loop must be replaced with semicolon.

3. The statement **%count** must be replaced with **%d** in the last line.

Q5. Write down the output of the following code segment.

a.

```
int sum = 0, p ;
for ( p =5; p <=25; p = p + 5)
        sum = sum + 5;
printf ("Sum is %d ", sum );
```

Ans: **Sum is 25**

b.

```
int i ;
for ( i = 34; i <= 60; i = i * 2 )
        printf( " * " );
```

Ans: *****

c.

```
for (int i = 50; i <= 50; i++)
{
        for (j =i; j >= 48; j--)
                printf ("j = %d\n", j);
        printf("i = %d\n ", i);
}
```

Ans: **j = 50**
j = 49
j = 48
j = 50

d.

```
int i, arr[] ={ 2, 3, 4, 5, 6, 7, 8 };
for (i = 0; i<7; i ++)
{
        printf("%d\n", arr [i] * arr [i]);
        i++;
}
```

Ans: **4**
16
36
64

e.

```
int i, j;
float ar1[] = {1.1, 1.2, 1.3};
```

```
float ar2[] = {2.1, 2.2, 2.3};
for(i=0; i<3; i++)
        for(j=i; j<3; j++)
                printf(" %f\n ", arl [i] * ar2[j] * i * j );
```

Ans: **0.000000**
0.000000
0.000000
2.640000
5.520000
11.959999

Unit 5: Functions

Exercise Solutions

Q1. Multiple Choice Questions

1) Functions could be built-in or.
a. Admin defined b. server defined
c. User defined d. Both a and c

2) The functions which are available in C Standard Library are, called
a. User-defined b. Built-in
c. Recursive d. Repetitive

3) The values passed to a function are called
a. Bodies b. Return types
c. Arrays d. Arguments

4) char cd() {return 'a'} . In this function "char" is
a. Body b. Return type
c. Array d. Arguments

5) The advantages of using functions are
a. Readability b. Reusability
c. Easy debugging d. All

6) If there are three return statements in the function body, of them will be executed.
a. One b. Two
c. Three d. First and Last

7) Readability helps to _________ the code.
a. Understand b. Modify c. Debug d. All

8) ________ means to transfer the control to another function.
a. Calling b. Defining c. Re-writing d. Including

Answers

1. c	2. b	3. d	4. b
5. d	6. a	7. d	8. a

Q2. Define the following terms.

1) Functions

Function

A **function** is a block of statements that performs a particular task. A program can have many functions. Each function in the program must have a unique name.

2) Built-in functions

A type of function that is available in C Standard Library is called **built-in function** or **library function**. These functions are predefined in C language. Built-in functions make programming faster and easier.

3) Functions Parameters

The variables in the function definition that receive the values are called **parameters**. The values of arguments are copied to parameters when a function is called. The function does not make any change to the values of the arguments.

4) Reusability

The code in a function is written once but can be reused many times. It allows the user to execute the same statements multiple times without writing it again. Suppose a function Line displays a line on the screen. It can be executed in different parts of the program to display the line again and again.

5) Calling a function

The statement that activates a function is known as function call. A function is called with its name. When a function is called, the following steps take place:

1. The control moves to the function that is called.
2. All statements in the function body are executed.
3. The control returns back to the calling function.

When the control returns back to the calling function, the remaining statements in the calling function are executed.

Q3. Briefly answer the following questions:

1. What is difference between arguments and parameters? Give an example.

Ans: The values that are passed to a function are called arguments. The variables in the function definition that receive the values are called parameters. The values of arguments are copied to parameters when a function is called.

```
void fun(int a)
{
        printf("%d", a);
}
void main()
{
        fun(5);
}
```

In the above example, the value 5 in function call is the argument. The variable **a** in the function **fun** is the parameter.

2. Enlist the parts of a function definition.

Ans: Two parts of a function definition include the function header and function body. The function header indicates the return-type, function-name and arguments. Function body contains the set of statements which are executed in the function.

3. Is it necessary to use compatible data types in function definition and function call? Justify your answer with an example.

Ans: Yes, it is necessary to use compatible data types in the function definition and the function call. The program may produce wrong output if the data types in function definition and function call are different.

```
void fun(int a)
{
        printf("%d", a);
}
void main()
{
        fun(2.5);
}
```

In the above example, the data type in function definition is **int** but the function call contains a **float** data. The program will display the value 2 instead of 2.5.

4. Describe the advantages of using functions.

Ans: See Long Q

5. What do you know about the return keyword?

Ans: The keyword return is used to return the value back to the calling function. The control moves back to the calling function along with the returned value when the return statement is executed in a function.

Q4. Identify errors in the following coding segments.

a.

```
void sum (int a, int b)
{
        return a + b ;
}
```

Ans: The return type of the function should be **int**.

b.

```
void message () ;
{
        printf("Hope you are fine : ) ");
        return 23;
}
```

Ans: 1. The semicolon at the end of first line is invalid.

2. The return type of the function should be **int**.

c.

```
int max (int a; int b)
{
        if (a > b)
                return a;
        return b;
}
```

Ans: The semicolon in the first line must be replaced with comma.

d.

```
int product ( int n1, int n2)
        return n1 * n2 ;
```

Ans: The statement **return n1 * n2;** must be enclosed in curly braces {}.

e.

```
int totalDigits ( int x)
{
        int count = 0 ;
        for (int i = x; i >=1, i= i/10)
                count++;
        return count
};
```

Ans: 1. The comma after i >= 1 in third line must be replaced with semicolon.
2. The semicolon after the function body must be removed.

Q5. Write down the output of the following code segment.

a.

```
int xyz (int n)
{
        return n + n;
}
int main( )
{
        int p = xyz(5);
        p = xyz(p);
        printf ("%d", p);
}
```

Ans: **20**

b.

```
void abc (int a, int b, int c)
{
        int sum = a+ b+ c;
}
int main ( )
{
```

```
        int x = 4, y =7, z =23, sum1=0;
        abc (x, y, z);
        printf( " %d %d %d " , x, y, z );
}
```

Ans: **4 7 23**

c.

```
int aa (int x)
{
        int p = x / 10;
        x++;
        p = p + (p * x);
        return p;
}
int main ( )
{
        printf("We got %d ", aa(aa(23)));
}
```

Ans: **We got 260**

d.

```
float f3(int nl, int n2)
{
        n1= n1 + n2;
        n2= n2 - n1;
        return 0;
}
int main ( )
{
        printf("%f\n ", f3(3,2));
        prinff("%f\n ", f3(10,6));
}
```

Ans: **0.000000**
0.000000